An Early CU Partition Mode Decision Algorithm in VVC Based on Variogram for Virtual Reality 360 Degree Videos

Mengmeng Zhang (yanhou@163.com)
Beijing Polytechnic College
Yan Hou
North China University of Technology
Zhi Liu
North China University of Technology

Research

Keywords: VVC, virtual reality, empirical variogram, intra coding, Mahalanobis distance

Posted Date: May 10th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-481775/v1

An Early CU Partition Mode Decision Algorithm in VVC Based on Variogram for Virtual Reality 360 Degree Videos

Mengmeng Zhang*[12], Yan Hou[2] and Zhi Liu*[2]

***Correspondence**:

Mengmeng Zhang: yanhou_email@163.com

Zhi Liu: lzliu@ncut.edu.cn

Yan Hou: 2754233008@qq.com

[1] Beijing Polytechnic College, Beijing 100144, China

[2] North China University of Technology, Beijing, 100144, China

Full list of author information is available at the end of the article.

Abstract

360-degree videos have become increasingly popular with the development of virtual reality (VR) technology. These videos are converted to a 2D image plane format before being encoded with standard encoders. To improve coding efficiency, a new generation video coding standard has been launched to be known as Versatile Video Coding (VVC). However, the computational complexity of VVC makes it time-consuming to compress 360-degree videos of high resolution. The diversity of CU partitioning modes of VVC greatly increases the computational complexity. Through statistical experiments on ERP videos, it is found that the probability of using horizontal partitioning for such videos is greater than that of vertical partitioning. The empirical variogram combined with Mahalanobis distance is proposed to measure texture orientation information. The experimental results show that the algorithm saves 32.13% of the coding time with only 0.66% BDBR increasing.
Key words: VVC, virtual reality, empirical variogram, intra coding, Mahalanobis distance.

1. INTRODUCTION

With the popularity of virtual reality applications, 360-degree video encoding has become a research hotspot. 360-degree video provides an immersive visual experience that can be viewed in all directions through a head-mounted display. These videos need to be converted into 2D images through projection transformation and compressed by standard encoders. Equirectangular projection (ERP) is the most commonly used format for 360-degree panoramic video projection. It maps spherical longitudes and latitudes to vertical and horizontal lines with constant spacing. The panoramic video contains information in all directions. In order to provide a realistic visual experience, most 360-degree videos are mainly presented with high resolutions of 4K, 6K and 8K. Therefore, an efficient encoder is essential.

Fig. 1. ERP projection schematic.

The Joint Video Experts Team (JVET) has developed a new generation of video coding standard Versatile Video Coding (VVC) [1]. H.266 / VVC uses a block-based hybrid video coding structure, which combines intra prediction, inter prediction, transform coding, and entropy coding. Its prediction, transformation, quantization, filtering, entropy coding, and other modules have been adjusted and improved compared with HEVC. VVC introduces a quadtree with nested multi-type tree (QTMT) block partitioning structure. A brief illustration of QTMT is shown in Fig.2. Traverse all possible partitioning modes and determine the best mode for the CU with minimum RD cost. Firstly, divide the CTU into quadtrees and partition the quad-leaf nodes with the multi-type tree structure. There are four types of partitions in the multi-type tree structure: vertical binary tree partition, horizontal binary tree partition, vertical ternary tree partition, and horizontal ternary tree partition. The encoding parameters limit the size of quadtrees and multi-type trees. For example, MaxQTSize and MaxMTSize limit the maximum root node size for quadtrees and multi-type trees, respectively. MinQTSize and MinMTSize limit the minimum root node size for quadtrees and multi-type trees, respectively. The concepts of PU and TU no longer exist in the QTMT structure. PU and TU are replaced by multi-type leaf nodes, and CU becomes the unit of prediction and transformation. Due to the existence of the QTMT structure, the size of CUs can range from the largest 128×128 to the smallest 4×4. The flexible CU significantly

improves compression efficiency. However, the increased flexibility comes at the cost of enlarging the search space and increasing the computational complexity.

A fast intra coding algorithm is proposed based on the characteristics of ERP videos and CU block partitioning under the VVC standard. Firstly, the CU partition mode of ERP video under QTMT partition structure is studied. Secondly, an effective early skip algorithm for the partition mode is proposed to reduce the computational burden of 360-degree video coding. The experimental results show that the algorithm can effectively reduce the coding complexity under the premise of ensuring coding performance.

The remainder of the paper is organized as follows. Section 2 presents the related work. Section 3 provides the statistics of CU partition mode, gives the motivation, and describes the proposed algorithm. The experimental results and conclusions are given in Sections 4 and 5, respectively.

Fig. 2. An illustration of QTMT structure: (a) CTU partition using QTMT; (b) The corresponding tree representation.

---

2. RELATED WORKS

Since the quadtree nested multi-type tree coding block structure is a major part of VVC, it has been studied extensively. However, there are few optimization algorithms for 360-degree video under VVC. A fast block partitioning algorithm for intra coding and inter coding is proposed in [2]. For intra coding, block-level Canny edge detectors are applied to extract edge features to skip the vertical or horizontal partition modes. For inter coding, the three-frame difference method is applied to determine whether the current block is a moving object or not and to terminate the partitioning in advance. The fast block partitioning algorithm based on Bayesian decision rules in [3] takes full advantage of the CU intra-mode and block partition information to advance the skipping of low probability partitioning modes. The QTBT partition decision algorithm in [4] strikes a balance between computational complexity and coding performance. At the CTU level, QTBT partitioning parameters are dynamically derived to accommodate local features. At the CU level, a joint classifier decision tree structure is designed to eliminate unnecessary iterations. A novel fast QTMT partitioning decision framework was developed in [5] based on the block size and coding pattern distribution characteristics. The fast intra coding partition algorithm based on variance and gradient proposed in [6] terminates the further splitting of smooth areas. QT partition is selected based on the gradient features extracted by the Sobel operator. Finally, by calculating the sub-library variance, one of the five possible QTMT partitioning modes is directly selected. Under the different spatial characteristics of the pixel domain, the intra coding CU partition fast decision algorithm in [7] implements the early determination of the binary tree partition mode.

For HEVC, the rapid determination of CU size is the focus of its research. Based on the structure tensor of each CU, an inter-mode decision algorithm is proposed in [8]. In [9], the time correlation between CU depth and coding parameters are applied to develop a selection model to estimate the range of candidate CU depths. Based on the spatiotemporal correlation, an adaptive depth range prediction method proposed in [10] reduces the complexity of HEVC coding. Researchers generally use fast coding algorithms based on texture features. The literature [11] proposes an adaptive fast mode

decision algorithm for HEVC intra coding based on texture features and multiple reference lines. According to the correlation between the CTU texture partition and the optimal CU partition, the number of recursive partitions of the CU is reduced in [12]. In [13], by analyzing the relationship between video texture and inter prediction mode, an inter-mode decision algorithm based on the texture correlation of adjacent viewpoints is proposed. In addition to manual optimization methods, a number of methods based on machine learning are used to reduce the complexity of HEVC. SVM is applied to determine the size of the CU in [14]-[16]. Fast algorithms based on the convolutional neural network (CNN) are proposed in [17]-[19].

Several studies have been conducted to reduce the coding complexity of 360-degree videos. The MPM is adapted in [20] to provide neighbor information accurately. Based on depth information, neighborhood correlation, and PU position, an adaptive algorithm is proposed to determine the best mode with less candidate RD-cost calculations. Besides, under the depth and SATD of the adjacent reference samples, early PU skipping and split termination are performed. Taking advantage of the fact that the prediction modes in the horizontal direction of the ERP video polar regions are selected more frequently than the remaining modes, a fast algorithm is designed in [21] to reduce the number of prediction modes evaluated in different regions.

3. PROPOSED ALGORITHM

The application of VVC multi-type trees has brought about a sharp increase in coding complexity, which further extends the coding time for 360-degree high-resolution videos. Most of the current fast partitioning algorithms are based on ordinary videos, and are not well applicable to 360-degree videos. According to the characteristics of ERP video CU partitioning, this paper designs a new index to measure the difference between the horizontal and vertical texture of CU, and proposes a fast CU partitioning algorithm based on empirical variogram to reduce coding complexity.

3.1. Observation and analysis

There is a lot of redundancy due to the ERP video stretching phenomenon, especially near the polar regions. Take the sequence DrivingInCity as an example.

Fig. 3. Partial block partition example of DrivingInCity: (a) High-latitude area; (b) Mid-latitude area; (c) The vehicle near the equator; (d) Buildings near the equator.

In Fig. 3, the blue line indicates quadtree partitions, the green line indicates horizontal binary tree partitions or horizontal ternary tree partitions, and the red line indicates vertical binary tree partitions or vertical ternary tree partitions. Intuitively, owing to the nature of the ERP video, the mid-latitude and high-latitude areas in Fig.3 (b) and Fig.3 (a) tend to use horizontal partitions or quadtree partitions, so they adopt CUs with relatively large size. The texture of large CUs is simple. Vertically partitioned CUs are small, and textures for small CUs are relatively complex. The partition modes near the equator are closely related to the image texture orientation. The texture of the vehicle in Fig.3 (c) tends to be horizontal, with CUs using more horizontal partitions than vertical partitions. The buildings in Fig.3 (d) has a vertical texture and thus use more vertical partitions than the vehicle in Fig.3 (c). To design an intra coding algorithm with low complexity, this paper experimentally explores the partitioning characteristics of ERP video coding and

computes the proportion of each partition mode. The experiments were conducted on VTM-4.0-360Lib-9.0. Under the Common Test Conditions, the sequence is encoded with the All-Intra configuration. The parameters are shown in Table 1.

Table 1 QTMT parameter settings.

CTU size	128×128
Maximum QT Depth	4
Minimum QT CU Size	8×8
Maximum MT Size	32×32
Maximum MT Depth	3
Minimum MT CU Size	4×4

This section conducts coding experiments on the 360-degree video sequence given by JVET under 4 kinds of QP and counts the partition pixels of the CU, and displays the 4K, 6K, and 8K sequences respectively, as shown in Fig.4. Where QT represents the quadtree partition, BT_H, BT_V, TT_H, and TT_V respectively represent four partition modes: horizontal binary tree, vertical binary tree, horizontal ternary tree, and vertical ternary tree.

Fig. 4. Pixel ratio of each partition: (a) 4K; (b) 6K; (c) 8K.

In terms of pixel ratios, horizontal CUs are 12.35% more than vertical CUs on average. For all given sequences, the CU using horizontal partitioning occupies more pixels than the CU using vertical partitioning, accounting for more than 30% of the multi-type tree partition. The sum of pixels using quadtree partitions and horizontal partitions accounts for about 87% of the total pixels. Therefore, the characteristics of the 360-degree video and the VVC partition method can be fully utilized to accelerate the speed of CU block partitioning.

3.2. Method for fast CU partition decision

The QTMT structure contains QT partitioning, BT partitioning, and TT partitioning. Each CU will choose the best mode with the lowest RD cost between 5 partitioning modes and no partitioning. Each mode will be traversed during RDO, which is a very time-consuming process. This paper attempts to predict the CU partition mode in advance by combining the features of ERP videos and VVC horizontal and vertical partitioning to skip the unnecessary RD cost calculation.

In recent studies of ERP video coding, algorithms that optimize the intra coding angle modes or early termination of CU partitions are usually applied to reduce complexity. Due to the nature of image stretching caused by ERP projection, the intra prediction mode between 2 and 18 in the angle prediction model are easier to be selected than other angles. However, the previous research was implemented in the HEVC standard. With the increase of 360-degree video data, the compression capability of HEVC cannot meet future needs. The partition mode using the only quadtree is not flexible enough for ERP video. To reduce the coding complexity under the condition of ensuring the coding quality, an accurate and fast CU texture direction discrimination method is required. Common texture discrimination methods include statistical and model methods, among which the typical methods are to use the gray level co-occurrence matrix (GLCM) and the Markov Random Field (MRF) model. Despite its adaptability and robustness, the GLCM

has a high computational complexity, which limits its practical application. The use of the MRF model requires hundreds of iterations and is therefore computationally intensive. In addition, image edge detection or gradient-based detection can be applied to determine textures. Edge detection emphasizes image contrast and detects luminance differences. The target boundary is the step change in luminance level and the edge is the location of the step change. Edge locations can be detected using first-order differentiation. Commonly used first-order edge detection operators mainly include Sobel operator and Canny operator. They can pre-determine the image features for obvious edge areas, but have limitations for large flat areas where the edge features are not obvious. Due to the high resolution of 360-degree videos, such flat areas are often present when showing water, sky and other environments.

In this paper, an early decision algorithm for CU partition modes is designed based on the idea of the variogram. In the ERP projection format, straight lines parallel to the spherical latitudes unfold into rows of rectangular planes, and texture stretching is evident at the poles. It is found through previous statistics that the texture stretching regions tend to be partitioned using a combination of horizontal binary trees, horizontal ternary trees, and quadtrees. The empirical variation function is simple to calculate, can effectively reduce the computational complexity, and can choose horizontal and vertical directions, which is more conducive to the texture similarity of the two directions. First, the Mahalanobis distance and the empirical variation function are used to calculate the function values in the horizontal and vertical directions, and then the selection range of the final partition mode is determined according to the degree of difference between the two directions, so as to realize the rapid selection of the CU partition mode.

Variogram has been used in texture analysis for many years [22]-[23]. It can adequately reflect the randomness and structure of image data. The theory of variogram considers not only the randomness of regionalized variables but also the spatial characteristics of data. The image data is not a purely random variable, it has obvious structural features, and the image pixels can be considered as a regionalized variable $Z(x)$. The two-point variogram describes the statistical characteristics of two points in the image space. Therefore, different textures have different values of the variogram, and the variogram can be applied to texture classification.

Let Z be a random function in the sampling space, x and t be the spatial position and step size, respectively. Assuming that the random function is second-order stationary, the variogram is defined as [24].

$$r(t)=\frac{1}{2}Var\left[Z(x)-Z(x+t)\right] \tag{1}$$

Where $r(t)$ represents the semi-variogram of the random function, and the semi-variogram is now referred to as the variogram to simplify technical jargon. In practical applications, empirical variogram $r(t)$ is expressed as:

$$r(t)=\frac{1}{2N(t)}\sum_{i=1}^{N(h)}\left[Z(x_i)-Z(x_i+t)\right]^2 \tag{2}$$

Where step t is the distance between two points in a certain direction and $N(t)$ is the number of all pairs of points that are two points away from t. When the value of t is 1, it is called a one-step variogram in this paper.

The Mahalanobis distance is used to measure the similarity between two unknown sample sets. It

differs from the Euclidean distance in that it takes into account the association between various features and normalize the covariance to make the relationship between the features more realistic. And it can better reflect the relationship between CU's rows or CU's columns and CU itself. The rows and columns of pixels are considered as sample sets. The difference between the rows and columns of CU was measured with Mahalanobis distance. This strategy has better noise-resistant. Set r_n as the transpose of the row vector. The Mahalanobis distance between two rows is defined in equation (5). in equation (4) represents the column vector.

$$r_n = \begin{pmatrix} x_{n1} \\ x_{n2} \\ \mathrm{M} \\ x_{nw} \end{pmatrix} \tag{3}$$

$$S_r = \begin{pmatrix} \mathrm{cov}(X_1, X_1) & \mathrm{cov}(X_1, X_2) & \mathrm{L} & \mathrm{cov}(X_1, X_w) \\ \mathrm{cov}(X_2, X_1) & \mathrm{cov}(X_2, X_2) & \mathrm{L} & \mathrm{cov}(X_2, X_w) \\ \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} \\ \mathrm{cov}(X_w, X_1) & \mathrm{cov}(X_w, X_2) & \mathrm{L} & \mathrm{cov}(X_w, X_w) \end{pmatrix} \tag{4}$$

$$D_M(r_i, r_j) = \sqrt{(r_i - r_j)^T S_r^{-1} (r_i - r_j)} \tag{5}$$

Set c_n as the column vector. The Mahalanobis distance between two columns is defined in equation (8). Y_n in equation (7) represents the row vector.

$$c_n = \begin{pmatrix} x_{1n} \\ x_{2n} \\ \mathrm{M} \\ x_{hn} \end{pmatrix} \tag{6}$$

$$S_c = \begin{pmatrix} \mathrm{cov}(Y_1, Y_1) & \mathrm{cov}(Y_1, Y_2) & \mathrm{L} & \mathrm{cov}(Y_1, Y_h) \\ \mathrm{cov}(Y_2, Y_1) & \mathrm{cov}(Y_2, Y_2) & \mathrm{L} & \mathrm{cov}(Y_2, Y_h) \\ \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} \\ \mathrm{cov}(Y_h, Y_1) & \mathrm{cov}(Y_h, Y_2) & \mathrm{L} & \mathrm{cov}(Y_h, Y_h) \end{pmatrix} \tag{7}$$

$$D_M(c_i, c_j) = \sqrt{(c_i - c_j)^T S_c^{-1} (c_i - c_j)} \tag{8}$$

According to equations (2), R_h and R_v are calculated from equation (9) and equation (10), respectively.

$$R_h = \frac{1}{2(w-1)} \sum_{j=1}^{w-1} D_M(c_j, c_{j+1})^2 \tag{9}$$

$$R_v = \frac{1}{2(h-1)} \sum_{i=1}^{h-1} D_M(r_i, r_{i+1})^2 \tag{10}$$

Fig. 5 and Fig. 6 are illustrations of R_h and R_v, respectively. To verify the efficiency of the strategy for texture recognition, the sequence DrivingInCity is used as an example to calculate the texture correlation in each CU distribution of R_h and R_v.

Fig. 5. Illustration of R_h in horizontal direction.

Fig. 6. Illustration of R_v in vertical direction.

The smaller the value of R_h and R_v, the greater the similarity of the textures in the corresponding directions. As the QP increases, the probability of $R_h < R_v$ increases as well (Fig. 7). Statistically, when the QP is 32, there are 30% CUs with $R_v \le R_h$, most of these CUs are distributed near the equator. And 70% CUs with $R_h < R_v$, these CUs are mainly distributed in the polar regions. It shows that the method can effectively distinguish the textures of the CU, and the ERP video has a high similarity in horizontal textures.

Fig. 7. Comparison of R_h and R_v.

R_h and R_v are utilized to determine the horizontal and vertical textures of CUs and adjusts the appropriate thresholds for different CUs based on their statistical properties to jump the over unnecessary RDO processes. Considering the setting of the QTMT parameters in Table 1, the size of the maximum multi-type tree is 32×32, so 32×32 pixel blocks are selected as the foundation for classification. Through Fig. 5, it is found that the utilization rate of the binary tree partition is much higher than that of the ternary tree, therefore, the sub-blocks of the two binary tree partitions of the 32×32 block are evaluated, and consider adding it's 32×16 and 16×32 sub-blocks to the algorithm for efficiency. These two CUs are obtained with the QT depth of 2 and the MT depth of 1. At this point, the quadtree partition has ended, so these two CUs will no longer perform quadtree partitioning. Including the case of no partition, there are only five partition modes in total. The 32×16 and 16×32 blocks are derived from horizontal and vertical partitions, respectively. Therefore, for 32×16 blocks, the non-partitioned and horizontally partitioned cases are discussed together. Similarly, for 16×32 blocks, the cases of non-partitioning and vertical partitioning are combined and discussed together. Through experimental statistics, it is found that in the process of encoding, the number of 32×16 blocks is about twice that of 16×32 blocks. About 83% of 32×16 blocks use horizontal partitioning, and about 68% of 16×32 blocks use vertical partitioning. Since the former are derived from horizontal division, they tend to adopt horizontal partitions. Similarly, the latter are derived from vertical division, so they tend to adopt vertical partitioning, but this tendency is weakened due to the stretching of the ERP video. In general, these two non-square blocks have good texture inheritance properties, and are easy to judge the texture. Since the overall number of samples is required to be

greater than the number of dimensions of the samples during the calculation of the Mahalanobis distance, the Euclidean distance is used for the case of 32×16 CU to calculate the vertical direction and 16×32 CU to calculate the horizontal direction. As for the 16×16 blocks, they are usually distributed in regions with complex textures, so they contain more information. Small-size CUs near the equator are more densely distributed and therefore less redundant. If the fast algorithm of this paper is applied to such blocks, reducing the same encoding complexity would result in more performance loss.

.

3.3. Selection of thresholds

Definition λ to measure the difference in texture between the horizontal and vertical directions

$$\lambda = \left| R_h - R_v \right| \tag{11}$$

For 32×16 and 16×32 CUs, since their child CUs share part of the content of the parent CUs, it is also possible to share the partition mode [3]. Therefore, the 32×16 CUs have a probability of using horizontal partitioning over vertical partitioning, and the probability of the 16×32 CUs using vertical partitioning is higher than that of horizontal partitioning. This feature can be adjusted through thresholds.

To choose a reasonable threshold, the accuracy and discrimination rate of the algorithm under different thresholds are counted. The definitions of the accuracy rate P_c and the discrimination rate P_d are as follows.

$$P_c = \frac{N_{skip} + N_c}{N_{total}} \tag{12}$$

$$P_d = \frac{N_d}{N_{total}} \tag{13}$$

Where N_{skip} represents the number of CUs using the original VTM algorithm when $\lambda < threshold$ and N_c indicates the number of CU when $R_h < R_v$ without vertical partition plus the number of CU when $R_h \geq R_v$ without horizontal partition in case $\lambda \geq threshold$. N_d represents the number of CUs with $\lambda \geq threshold$, and N_{total} represents the total number of CU.

Fig. 8. P_c and P_d at different thresholds.

Through statistics, it is found that P_c is about 60% when the threshold is set to 0. With the increase of the threshold, P_c gets close to 100%, and P_d decreases accordingly. As shown in Fig. 8, when it is set to 0.6, P_c and P_d are both about 70%. When it is set to 0.8, P_c can reach 85%, and P_d at this point is 60%. When it is set to 1.2, P_d drops to 25%, and P_c increases to 96%.

To balance coding speed and performance, different thresholds are used for different CUs. The reference threshold for the 32×32 CU is set to 0.8. Since the 32×16 CU has a high probability of using the horizontal partitioning, a high discrimination rate can effectively reduce the coding complexity, so the threshold for the 32×16 CU is set to 0.6. Although the probability of vertical

partitioning is greater than horizontal partitioning for 16×32 CUs, there are still many CUs that adopt horizontal partitioning due to the influence of the projection format. To ensure the accuracy of the algorithm, the threshold for the 16×32 CU is set to 1.2. There is also a part of the CUs that have a higher probability of vertical partitioning, and the threshold is set to 1.2 to skip the horizontal partitioning under the premise of ensuring correctness. The procedure is shown in Fig. 9.

For a CU with size 32×32, if $R_h < R_v$ and $\lambda \geq 0.8$, it indicates that the texture similarity of the CU in the horizontal direction is higher than that in the vertical direction and skips the vertical partitioning of the current block. For a 16×32 CU block, if $R_h < R_v$ and $\lambda \geq 1.2$, it proves that the CU has a high texture similarity in the horizontal direction, skipping the current vertical partitioning. For a 32×16 CU block, if $R_h < R_v$ and $\lambda \geq 1.2$, it shows that the CU horizontal textures are similar enough to skip the vertical partitioning of the current block. In other cases, the original VTM is used for encoding.

Fig. 9. The Flow chart of the proposed fast partition algorithm.

---

4. EXPERIMENTAL RESULTS AND DISCUSSION

Experimental results are presented in this section. The algorithm is implemented in the VVC test model VTM-4.0-360Lib-9.0. The 360 test sequences are encoded using the All-Intra configuration under the Common Test Conditions, with QP values of 22, 27, 32, and 37. Using BDBR, time-saving ΔTS, and BDPSNR to evaluate the performance of the algorithm. ΔTS is defined as follows.

$$\Delta TS(\%) = \frac{1}{4}\sum_{i\in Q}\frac{T_{VTM}(i) - T_p(i)}{T_{VTM}(i)} \times 100\% \tag{14}$$

Where $T_{VTM}(i)$ and $T_p(i)$ represent the total encoding time of the reference VTM encoders and the proposed algorithm under QP i, respectively. Q represents the QP set with {22, 27, 32, 37} in this paper.

In order to verify the validity of the algorithm, the proposed algorithm was used for ordinary sequences and compared with that of Yang [5]. As seen in Table 2, with the proposed algorithm, the encoding time is reduced by 38.33% while the increase of BDBR is only 0.81%. Yang's algorithm achieves a time saving of 52.69% while increasing the BDBR by 1.78%, in some cases, less BDBR loss is required.

Table 2. Performance comparison with Yang's algorithm

Class	*Sequences*	Yang		Proposed	
		ΔTS(%)	*BDBR(%)*	*ΔTS(%)*	*BDBR(%)*
B	*Kimono*	63.87	1.90	37.93	0.66
	ParkScene	56.60	1.33	37.78	0.72
	Cactus	56.66	1.95	38.53	0.99
	BasketballDrive	64.01	2.25	43.49	0.96
	BQTerrace	56.07	2.07	35.55	0.69
C	*BasketballDrill*	48.19	2.01	40.66	1.35

	BQMall	55.23	2.15	39.09	0.87
	PartyScene	45.73	0.60	34.36	0.20
	RaceHorsesC	48.39	1.16	39.98	0.85
D	*BasketballPass*	45.85	2.33	42.34	1.12
	BQSquare	46.06	0.81	29.24	0.13
	BlowingBubbles	41.56	0.77	40.44	0.40
	RaceHorses	43.17	0.86	40.57	0.70
E	*FourPeople*	57.64	2.75	41.29	1.23
	Johnny	58.98	3.29	41.18	1.60
	KristenAndSara	59.19	2.51	40.75	1.11
F	*BasketballDrillText*	47.66	1.82	38.29	1.19
	ChinaSpeed	52.67	1.30	35.93	0.49
	SlideEditing	48.91	1.12	33.53	0.31
	SlideShow	57.37	2.61	35.60	0.64
	Average	52.69	1.78	38.33	0.81

Table 3 shows the experimental results of the proposed algorithm in this paper. Compared with VTM4.0-360Lib-9.0, it can be observed that the coding complexity can be effectively reduced without significant fluctuations in the coding performance of all test sequences. The encoding time is reduced by 32.31% on average, and the BDBR is only averagely increases by 0.66%. The BDPSNR averagely decreases by 0.033dB. *SkateboardInLot* has the least time-saving and *Harbor* has the most.

Table 3. Experimental results of the algorithm proposed in this paper compared with VTM4.0-360Lib-9.0 encoder

Class	***Sequences***	***BDBR (%)***	***ΔTS(%)***	***BDWSPSNR (dB)***
8K	*ChairliftRide*	0.53	31.95	-0.025
	GasLamp	1.23	33.31	-0.057
	Harbor	0.94	35.56	-0.044
	KiteFlite	0.56	35.31	-0.035
	SkateboardInLot	0.65	29.95	-0.030
	Trolley	0.51	31.14	-0.032
6K	*Balboa*	0.68	30.33	-0.038
	BranCastle2	0.32	31.98	-0.020
	Broadway	1.18	32.20	-0.067
	Landing2	0.57	31.58	-0.029
4K	*AerialCity*	0.59	31.70	-0.020
	DrivingInCity	0.76	31.97	-0.031
	DrivingInCountry	0.36	30.59	-0.018
	PoleVault	0.41	32.25	-0.024
	Average	0.66	32.13	-0.033

To verify the coding performance of non-square blocks, the algorithm is adjusted to use only CUs with 16×32 and 32×16, the thresholds are set to 1.2 and 0.6, respectively. The experimental results

are shown in Table 4. Compared with the reference encoder, the encoding time averagely reduces by 22.42%. The BDBR increases by 0.44% on average. The BDPSNR averagely decreases by 0.022dB. It illustrates the effectiveness of the algorithm for non-square blocks.

Table 4. Experimental results of the algorithm proposed in this paper for 16×32 and 32×16 CUs compared with VTM4.0-360Lib-9.0 encoder

Class	***Sequences***	***BDBR (%)***	***ΔTS(%)***	***BDWSPSNR (dB)***
8K	*ChairliftRide*	0.32	25.75	-0.015
	GasLamp	0.80	23.28	-0.037
	Harbor	0.68	24.78	-0.032
	KiteFlite	0.41	25.48	-0.026
	SkateboardInLot	0.49	20.59	-0.023
	Trolley	0.36	22.64	-0.022
6K	*Balboa*	0.43	20.19	-0.024
	BranCastle2	0.25	21.16	-0.016
	Broadway	0.70	20.85	-0.040
	Landing2	0.38	21.31	-0.020
4K	*AerialCity*	0.37	23.51	-0.013
	DrivingInCity	0.47	21.22	-0.019
	DrivingInCountry	0.26	21.92	-0.013
	PoleVault	0.28	21.19	-0.016
	Average	0.44	22.42	-0.022

Considering the strategy of further reducing the coding complexity, it is verified experimentally whether 16×16 CUs are added to the algorithm. As with 32×32 CUs, the threshold is set to 0.8. Table 5 shows the experimental results after combining 16×16 CUs. It is observed that the average encoding time is reduced by 39.43%, meanwhile, the BD performance degradation is 0.96% on average. The average BDPSNR is decreased by 0.048dB. Compared with the previous algorithm, this algorithm saves more coding time. However, the BDBR of individual sequences such as GasLamp and Broadway has increased to more than 1.5%. In 8K video sequences, WS-PSNR has dropped a lot. Therefore, 16×16 CUs are not included in the algorithm of this paper.

Table 5. Experimental results of the algorithm proposed in this paper for 32×32, 16×32, 32×16 and 16×16 CUs compared with VTM4.0-360Lib-9.0 encoder

Class	***Sequences***	***BDBR (%)***	***ΔTS(%)***	***BDWSPSNR (dB)***
8K	*ChairliftRide*	0.74	37.57	-0.034
	GasLamp	1.60	38.00	-0.074
	Harbor	1.32	40.25	-0.061
	KiteFlite	0.78	42.84	-0.049
	SkateboardInLot	0.96	35.15	-0.044
	Trolley	0.71	38.82	-0.044

6K	*Balboa*	0.89	37.24	-0.049
	BranCastle2	0.52	41.63	-0.033
	Broadway	1.62	38.15	-0.091
	Landing2	0.84	40.26	-0.042
4K	*AerialCity*	0.90	40.18	-0.030
	DrivingInCity	1.14	39.49	-0.046
	DrivingInCountry	0.67	41.00	-0.032
	PoleVault	0.75	41.49	-0.043
	Average	0.96	39.43	-0.048

Fig. 10 compares the block partitions compressed by the anchor and the proposed scheme, respectively. It can be observed that the proposed scheme uses more horizontal partitions than VTM in the shaded area. Besides, there are a small number of unit blocks with different partition modes, but their final split modes are still similar.

Fig. 10. Comparisons of the block partitions between anchor (VTM4.0-360Lib-9.0) and the proposed scheme: (a) Ancher; (b) Proposed.

Fig. 11 shows the RD curves of the proposed algorithm and reference VTM4.0-360Lib-9.0 encoder in different sequences, of which (a) and (b) are 4K video sequences, (c) and (d) are 6K video sequences, (e) and (f) are 8K video sequences. It is observed that the performance of the proposed algorithm is similar to that of VTM4.0.

Fig. 11. Performance comparison between the proposed algorithm and VTM4.0-360Lib-9.0: (a) AerialCity; (b) DrivingInCity; (c) Balboa; (d) Broadway; (e) ChairliftRide; (f) GasLamp.

---

5. CONCLUSION

In this paper, a fast CU partition mode decision algorithm for ERP videos is proposed. According to the characteristics of ERP video, utilize R_h and R_v to determine the texture correlation between the two directions, and skip the horizontal or vertical partition modes in advance. Different thresholds are set to reduce encoding time while maintaining partition accuracy. Experimental results show that compared with VTM4.0-360Lib-9.0, the proposed algorithm saves 32.13% of the coding time, and the BD performance degradation is 0.66% on average.

Declarations

Abbreviations

VVC: Versatile Video Coding; HEVC: High Efficiency Video Coding; CU: Coding unit; VTM: VVC test model; BDBR: Bjontegaard delta bit rate; BDPSNR: Bjøntegaard delta peak signal-to-noise rate. ERP: Equirectangular projection; CTU: Coding tree unit; CU: Coding unit; JVET: Joint Video Exploration Team; LCU: Largest coding unit; PSNR: Peak to signal noise ratio; QP: Quantization parameters; RD: Rate distortion; RD cost: rate distortion cost; RDO: Rate-distortion operation; VR: Video reality; QT: Quadtree; BT: Binary tree; TT: Ternary tree; SVM: Support vector machine; CNN: Convolutional neural network; SATD: Sum of absolute transformed difference; WSPSNR: PSNR weighted by sample area; BDWSPSNR: BDPSNR weighted by sample area.

Availability of data and materials

The conclusion and comparison data of this article are included within the article.

Competing interests

The authors declare that they have no competing interests.

Funding: This work is supported by Great Wall Scholar Project of Beijing Municipal Education Commission (CIT&TCD20180304), Beijing Municipal Natural Science Foundation (No.4202018).

Authors' contributions

MZ proposed the framework of this work, and YH carried out the whole experiments and drafted the manuscript. ZL offered useful suggestions and helped to modify the manuscript. All authors read and approved the final manuscript.

Acknowledgements

Not applicable.

Authors' information

MMZ: Doctor of Engineering, professor, master instructor, master of Communication and Information Systems. His major research interests include the video codec, embedded systems, image processing, and pattern recognition. He has authored or co-authored more than 40 refereed technical papers in international journals and conferences in the field of video coding, image processing, and pattern recognition. He holds 21 national patents and 2 monographs in the areas of image/video coding and communications.

YH: Studying master of North China University of Technology. His major research is VVC.

ZL: Doctor of Engineering, master instructor. He received the B.S. degree in electronic information technology and the Ph.D. in signal and information processing from Beijing Jiaotong University, China in 2001 and 2011 respectively. Currently, he is a lecturer in North China University of Technology. His major research interests include the video codec, pattern recognition, and self-organizing network.

References

1. J. Chen, Y. Ye, and S. H. Kim, “Algorithm description for Versatile Video Coding and Test Model 4 (VTM 4).” Document JVET-M1002, Marrakech, Morocco, Jan. 2019.
2. N. Tang, J. Cao, F. Liang, J. Wang, H. Liu, X. Wang, and X. Du, “Fast CTU partition decision algorithm for VVC intra and inter coding.” 2019 IEEE Asia Pacific Conference on Circuits and Systems, Bangkok, Thailand, pp.361-364, Nov. 2019.
3. T. Fu, H. Zhang, F. Mu, and H. Chen, “Fast CU partitioning algorithm for H.266/VVC intra-frame coding.” 2019 IEEE International Conference on Multimedia and Expo, Shanghai, China, pp.55-60, Jul. 2019.
4. Z. Wang, S. Wang, J. Zhang, S. Wang, and S. Ma, “Effective quadtree plus binary tree block partition decision for future video coding.” 2017 Data Compression Conference, Snowbird, UT, USA, pp.23-32, Apr. 2017.
5. Y. Hao, S. Liquan, D. Xinchao, et al. “Low-Complexity CTU Partition Structure Decision and Fast Intra Mode Decision for Versatile Video Coding” IEEE Transactions on Circuits and Systems for Video Technology, vol. 30, no. 6, pp. 1668 - 1682, June 2020.
6. J. Chen, H. Sun, J. Katto, X. Zeng and Y. Fan, “Fast QTMT partition decision algorithm in VVC intra coding based on variance and gradient.” 2019 IEEE Visual Communications and Image Processing, Sydney, Australia, pp.1-4, Dec. 2019.
7. T. Lin, H. Jiang, J. Huang and P. Chang, “Fast binary tree partition decision in H.266/FVC intra coding.” 2018 IEEE International Conference on Consumer Electronics-Taiwan, Taichung, Taiwan, pp.1-2, May 2018.
8. S. Bakkouri, A. Elyousfi, and H. Hamout, “Fast CU size and mode decision algorithm for 3D-HEVC intercoding.” Multimedia Tools and Applications, vol.79, no.11-12, pp.6987-7004, Mar. 2020.
9. Y. Li, G. Yang, Y. Zhu, X. Ding, and X. Sun, “Adaptive inter CU depth decision for HEVC using optimal selection model and encoding parameters.” IEEE Transactions on Broadcasting, vol.63, no.3, pp.535-546, Sept. 2017.
10. Y. Kuo, P. Chen, and H. Lin, “A spatiotemporal content-based CU size decision algorithm for HEVC.” IEEE Transactions on Broadcasting, vol.66, no.1, pp.100-112, Mar. 2020.
11. R. Tian, Y. Zhang, and M. Duan, “Adaptive intra mode decision for HEVC based on texture characteristics and multiple reference lines.” Multimedia Tools and Applications, vol.78, no.1, pp.289-310, Jan. 2019.
12. W. Zhu, Y. Yi, and H. Zhang, “Fast mode decision algorithm for HEVC intra coding based on texture partition and direction.” Journal of Real-Time Image Processing, vol.17, no.2, pp.275-292, Apr. 2020.
13. J. Chen, B. Wang, J. Liao, and C. Cai, “Fast 3D-HEVC inter mode decision algorithm based on the texture correlation of viewpoints.” Multimedia Tools and Applications, vol.78, no.20, pp.29291-29305, Oct. 2019.
14. Y. Zhang, Z. Pan, N. Li, X. Wang, G. Jiang, and S. Kwong, “Effective data driven coding unit size decision approaches for HEVC intra coding.” IEEE Transactions on Circuits and Systems for Video Technology, vol.28, no.11, pp.3208-3222, Nov. 2018.
15. X. Liu, Y. Li, D. Liu, P. Wang and L. T. Yang, “An adaptive CU size decision algorithm for HEVC intra prediction based on complexity classification using machine learning.” IEEE

Transactions on Circuits and Systems for Video Technology, vol.29, no.1, pp.144-155, Jan. 2019.

16. M. Grellert, B. Zatt, S. Bampi and L. A. Da Silva Cruz, “Fast coding unit partition decision for HEVC using support vector machines.” IEEE Transactions on Circuits and Systems for Video Technology, vol.29, no.6, pp.1741-1753, Jun. 2019.
17. Z. Chen, J. Shi, and W. Li, “Learned fast HEVC intra coding.” IEEE Transactions on Image Processing, vol.23, pp.5431-5446, Mar. 2020.
18 Y. Li, Z. Liu, X. Ji and D. Wang, “CNN based CU partition mode decision algorithm for HEVC inter coding.” 2018 25th IEEE International Conference on Image Processing, Athens, Greece, pp.993-997, Oct. 2018.
19 S. Bouaafia, R. Khemiri, F. Sayadi, and M. Atri, “Fast CU partition-based machine learning approach for reducing HEVC complexity.” Journal of Real-Time Image Processing, vol.17, no.1, pp.185-196, Feb. 2020.
20 Y. Wang, Y. Li, D. Yang and Z. Chen, “A fast intra prediction algorithm for 360-degree equirectangular panoramic video.” 2017 IEEE Visual Communications and Image Processing, St. Petersburg, FL, USA, pp.1-4, Dec. 2017.
21 I. Storch, B. Zatt, L. Agostini, L. A. Da Silva Cruz and D. Palomino, “Fastintra360: A fast intra-prediction technique for 360-degrees video coding.” 2019 Data Compression Conference, Mar. 2019.
22 Tuan D. Pham, “The multiple-point variogram of images for robust texture classification.” 2016 IEEE International Conference on Acoustics, Speech and Signal Processing, Shanghai, China, pp.1303-1307, Mar. 2016.
23 Tuan D. Pham, “The semi-variogram and spectral distortion measures for image texture retrieval.” IEEE Transactions on Image Processing, vol.25, no.4, pp.1556-1565, Apr. 2016.
24 R.A. Olea, Geostatistics for Engineers and Earth Scientists, Kluwer Academic Publishers, Boston, 1999.
25 H. Yang, L. Shen, X. Dong, Q. Ding, P. An, and G. Jiang, “Low-Complexity CTU Partition Structure Decision and Fast Intra Mode Decision for Versatile Video Coding.” IEEE Transactions on Circuits and Systems for Video Technology, vol.30, no.6, pp.1668-1682, Jun. 2020.

Figure legends

Fig. 1. ERP projection schematic.

Fig. 2. An illustration of QTMT structure: (a) CTU partition using QTMT; (b) The corresponding tree representation.

Fig. 3. Partial block partition example of DrivingInCity: (a) High-latitude area; (b) Mid-latitude area; (c) The vehicle near the equator; (d) Buildings near the equator.

Fig. 4. Pixel ratio of each partition: (a) 4K; (b) 6K; (c) 8K.

Fig. 5. Illustration of R_h in horizontal direction.

Fig. 6. Illustration of R_v in vertical direction.

Fig. 7. Comparison of R_h and R_v

Fig. 8. P_c and P_d at different thresholds.

Fig. 9. The Flow chart of the proposed fast partition algorithm

Fig. 10. Comparisons of the block partitions between anchor (VTM4.0-360Lib-9.0) and the proposed scheme: (a) Ancher; (b) Proposed.

Fig. 11. Performance comparison between the proposed algorithm and VTM4.0-360Lib-9.0: (a) AerialCity; (b) DrivingInCity; (c) Balboa; (d) Broadway; (e) ChairliftRide; (f) GasLamp.

Figures

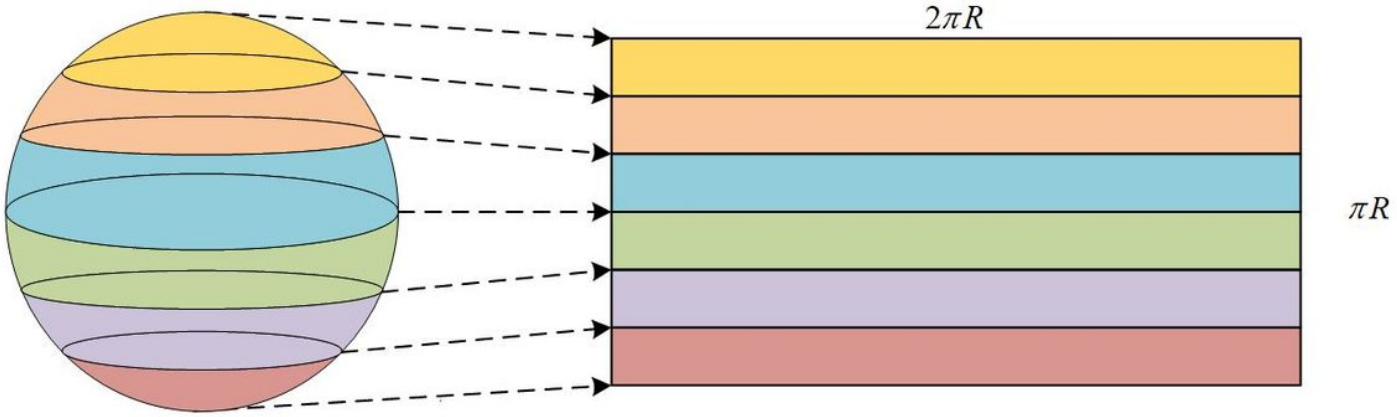

Figure 1

ERP projection schematic

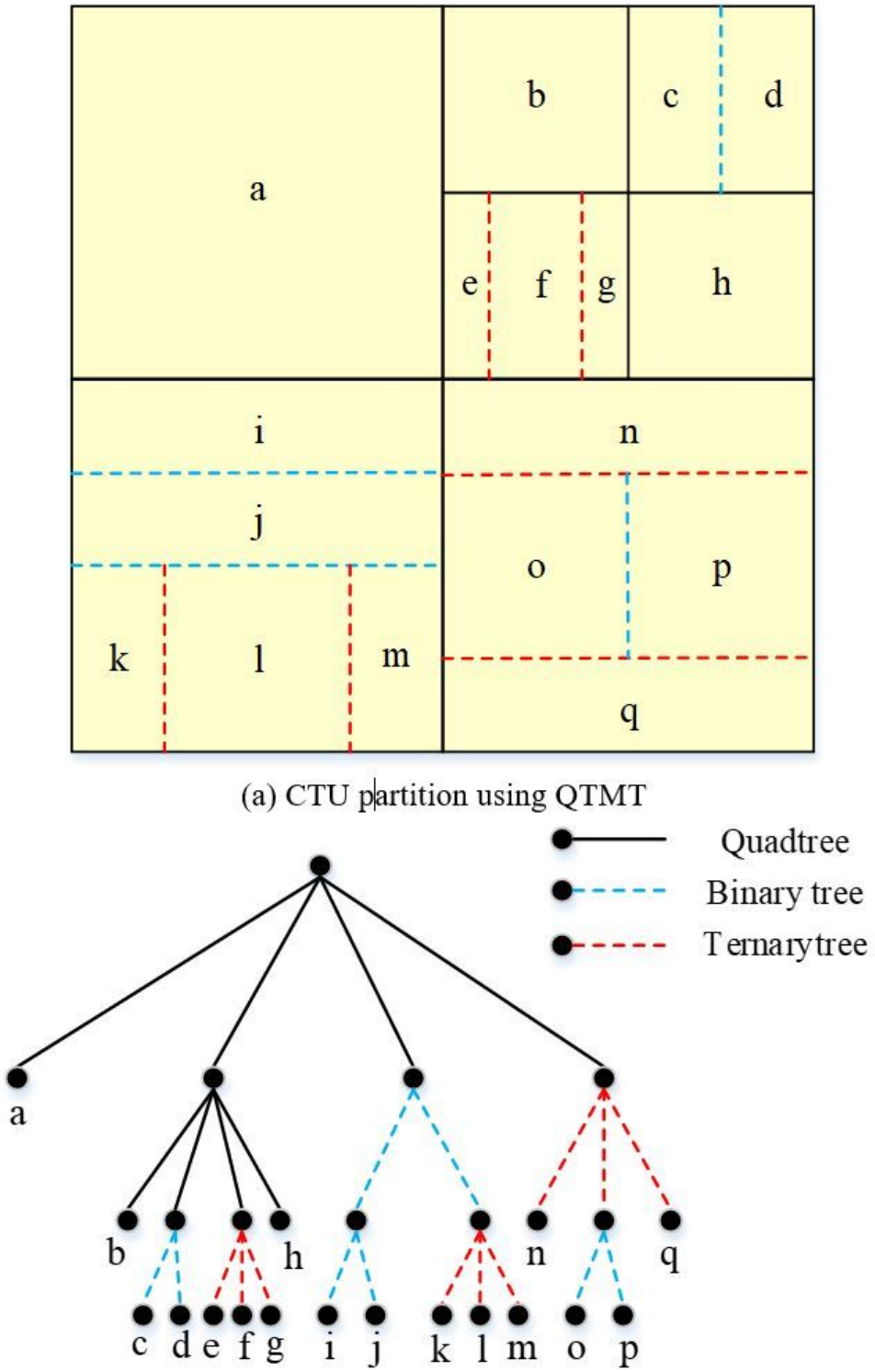

Figure 2

An illustration of QTMT structure

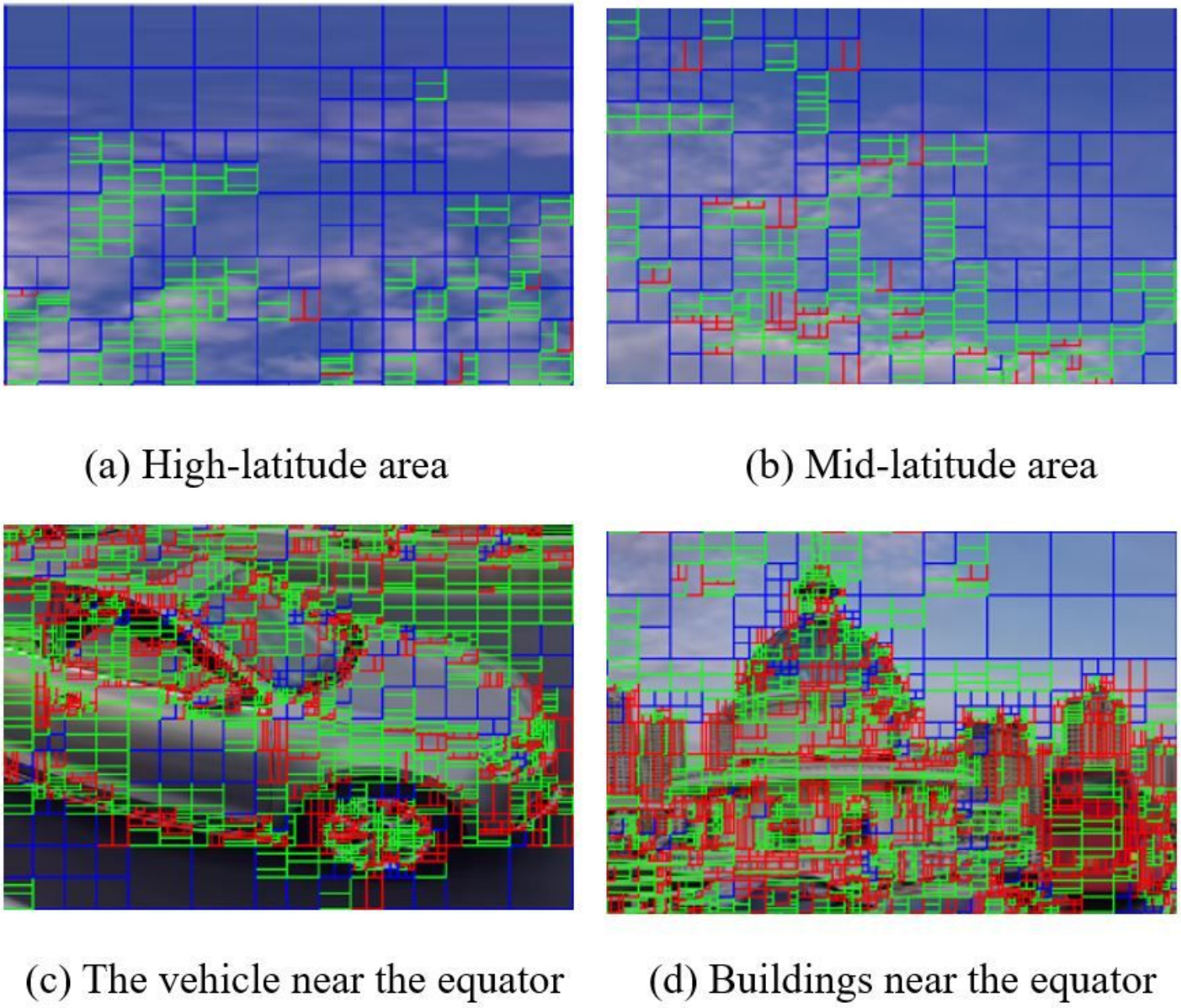

Figure 3

Partial block partition example of DrivingInCity.

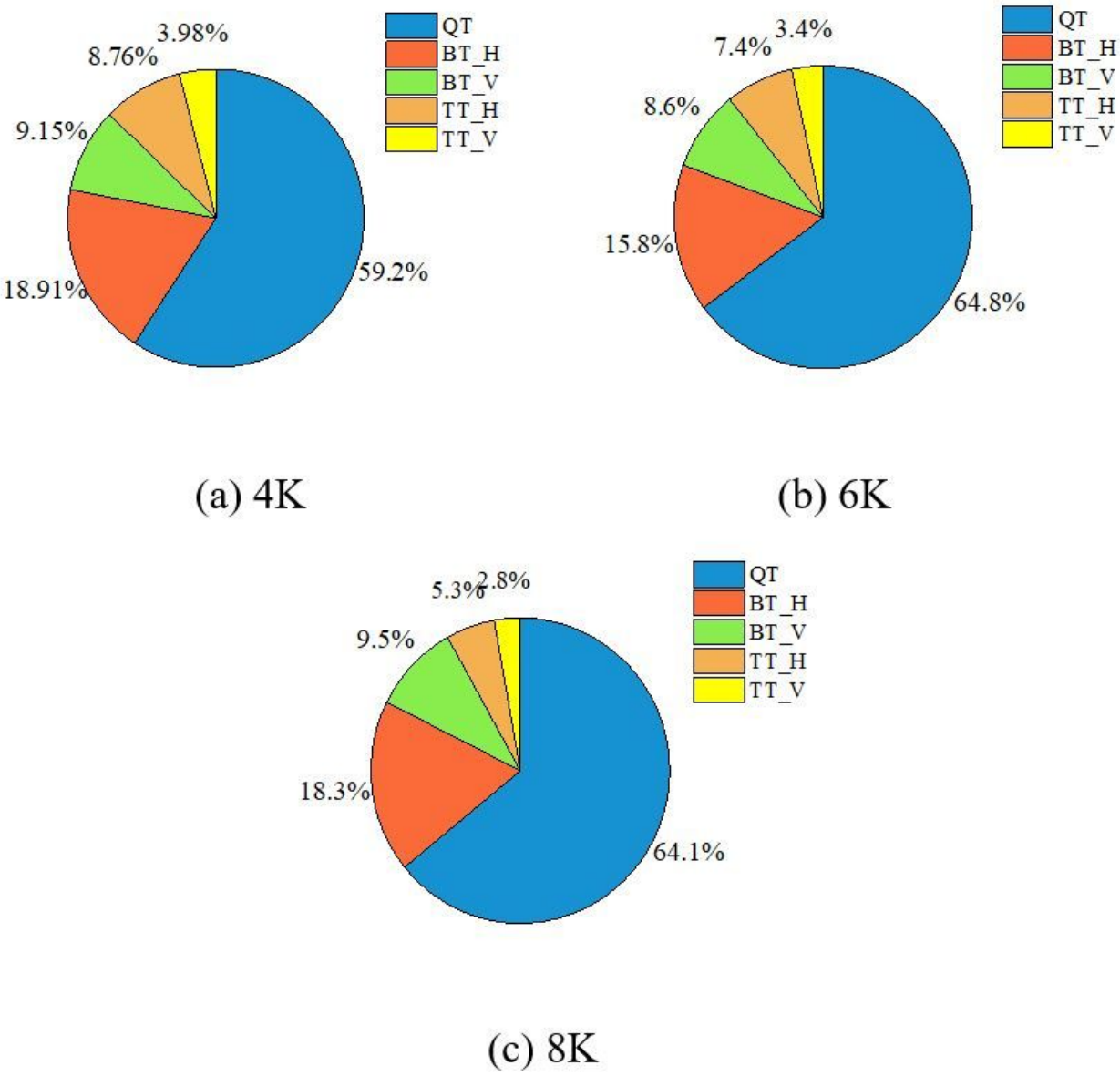

Figure 4

Pixel ratio of each partition

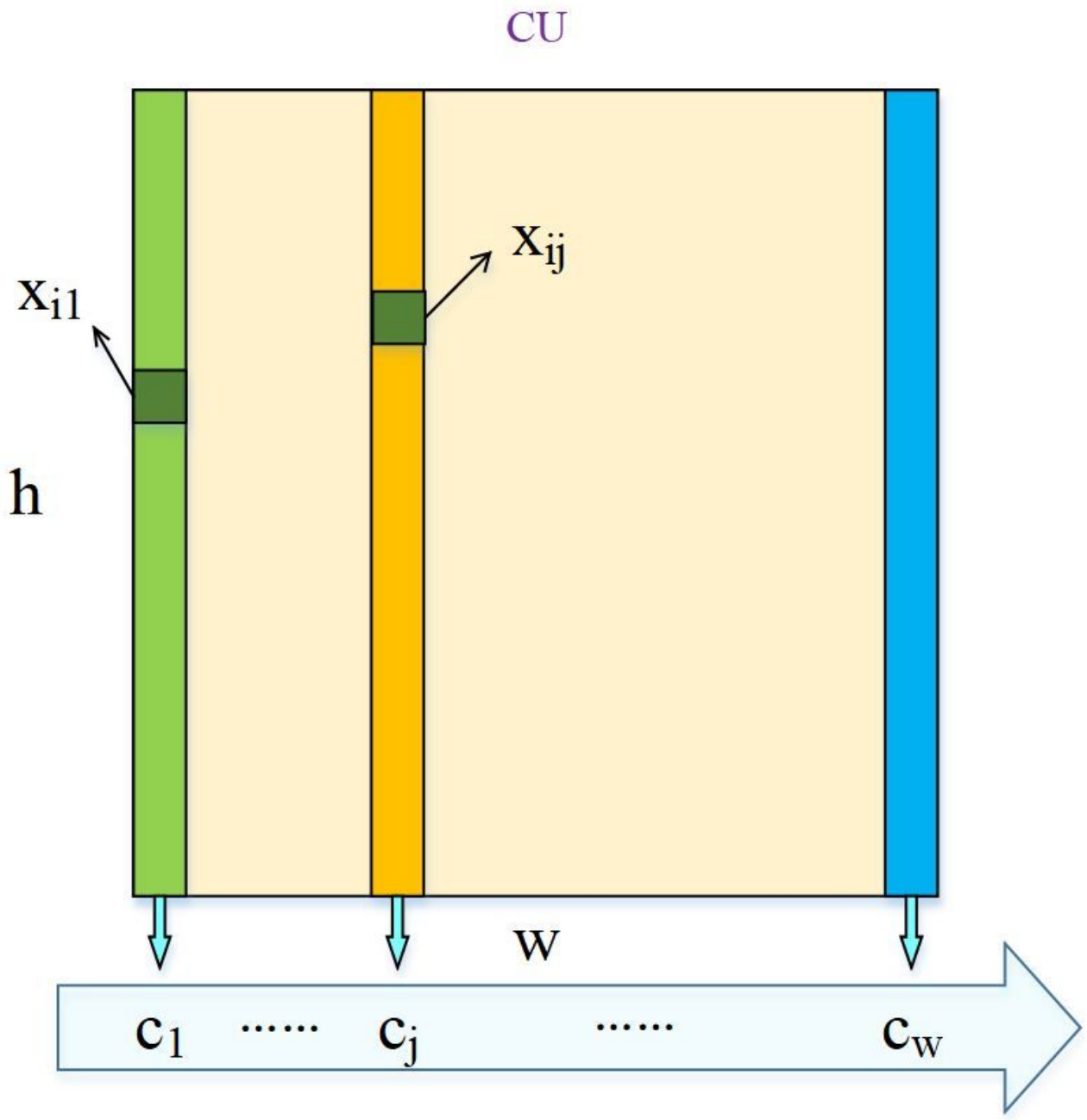

Figure 5

Illustration of Rh in horizontal direction.

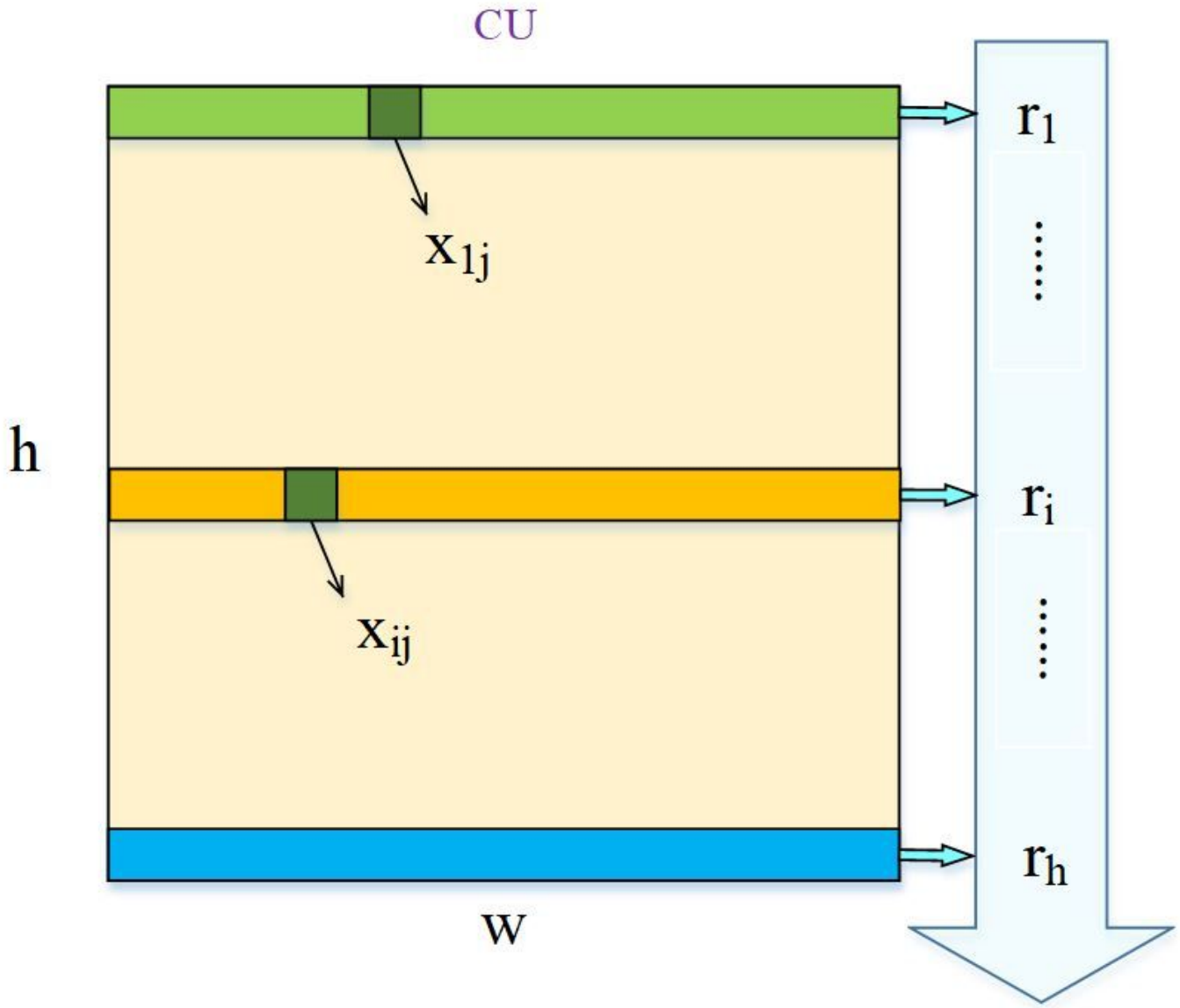

Figure 6

Illustration of Rv in vertical direction.

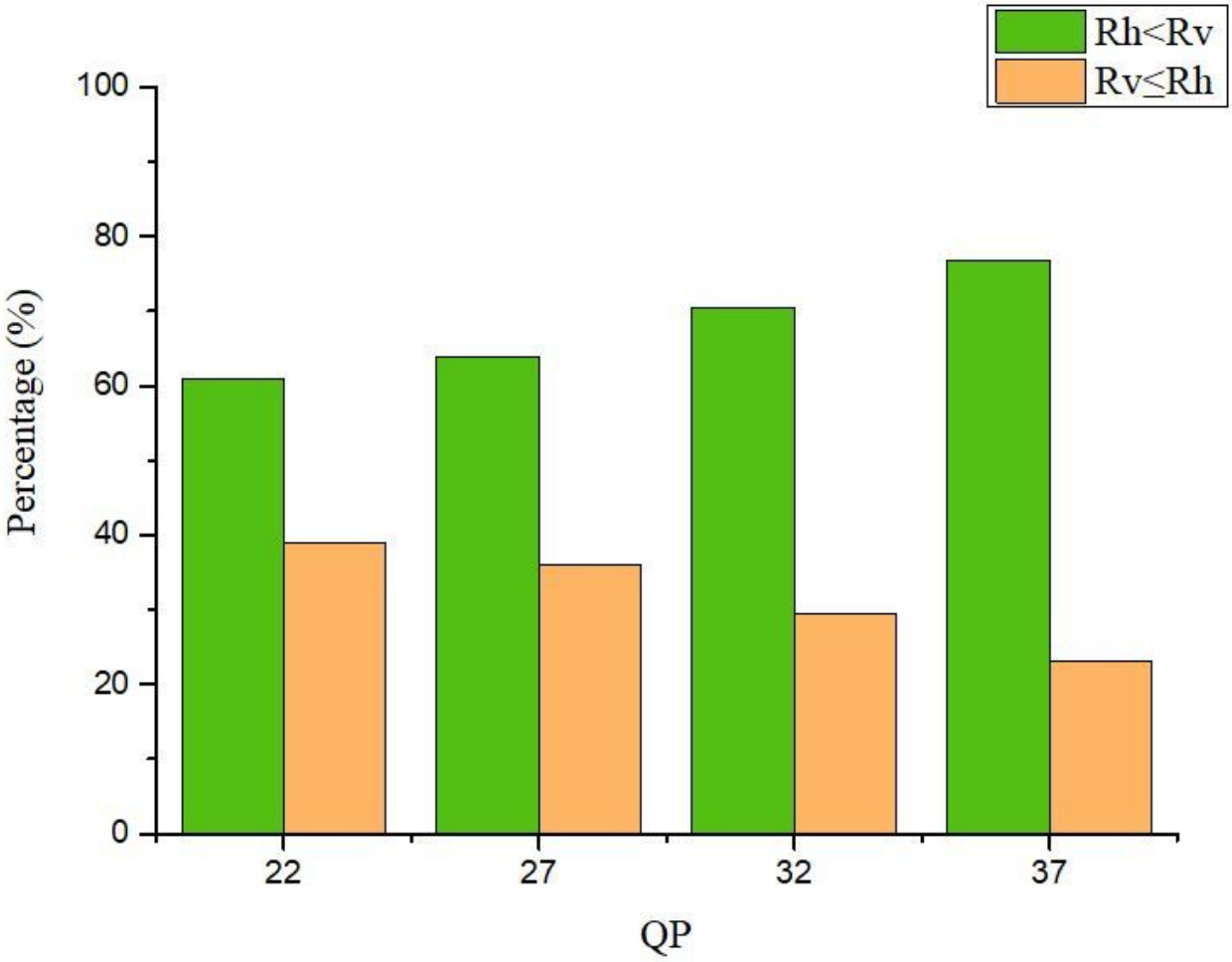

Figure 7

Comparison of Rh and Rv.

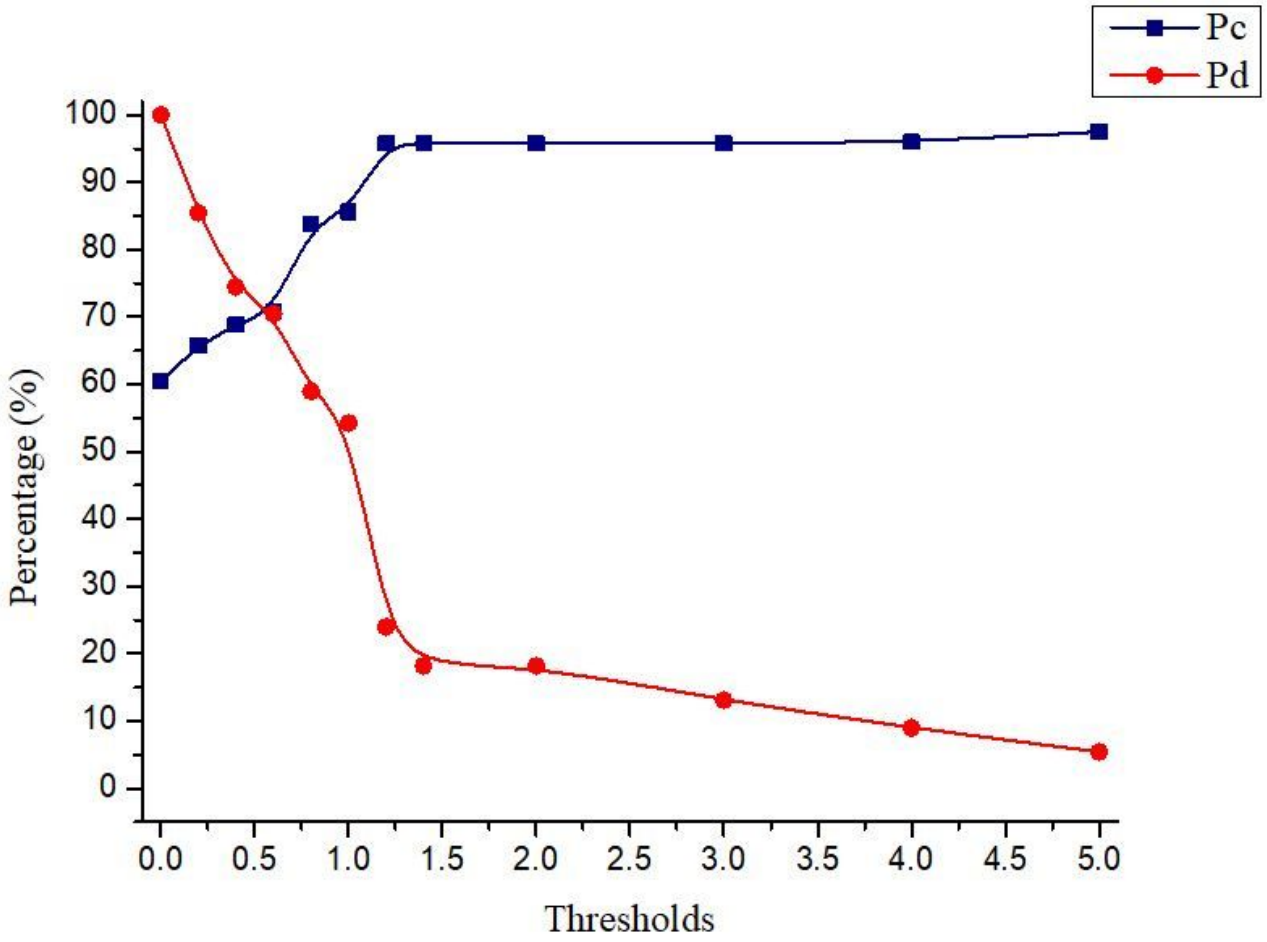

Figure 8

Pc and Pd at different thresholds.

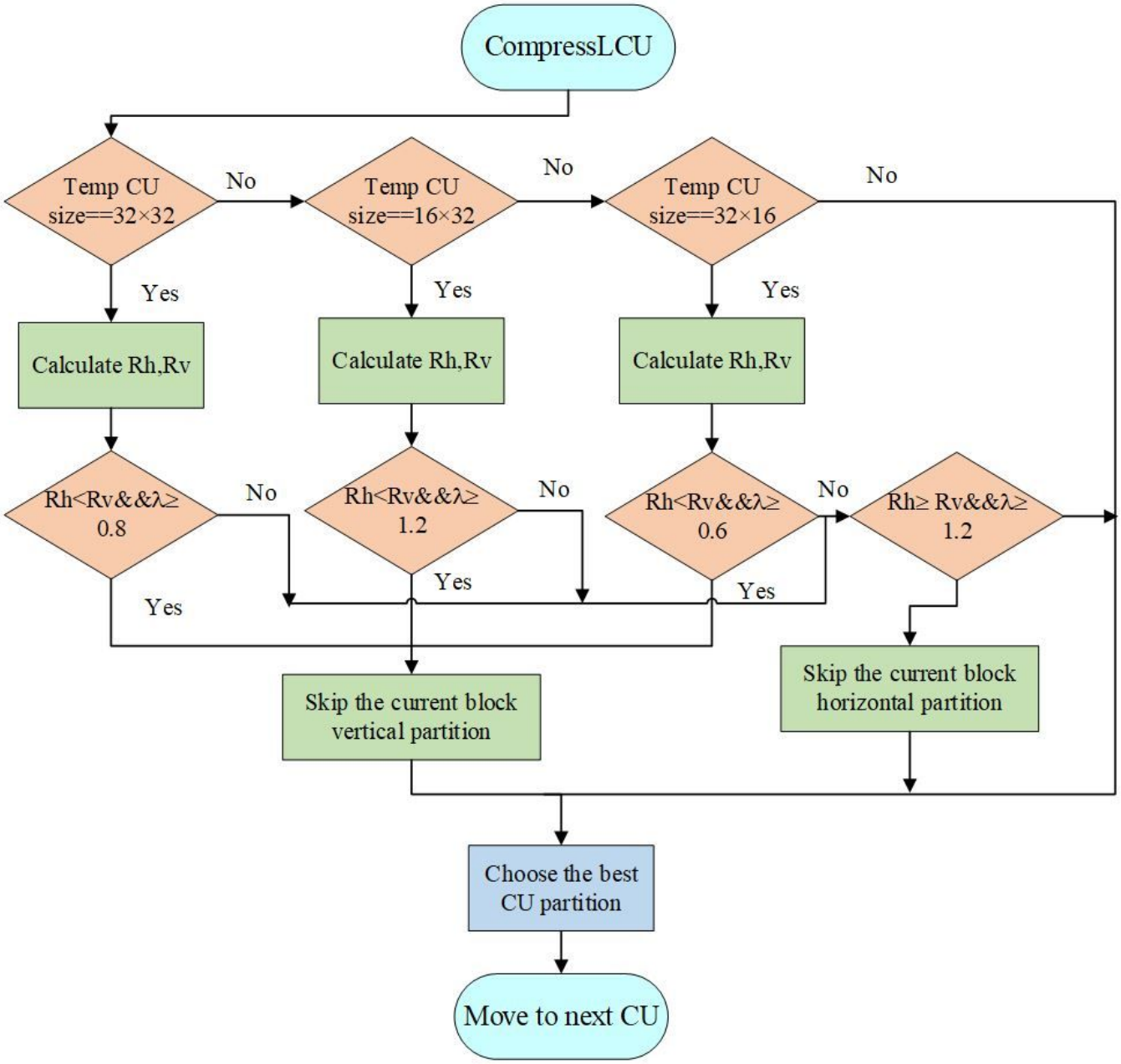

Figure 9

The Flow chart of the proposed fast partition algorithm.

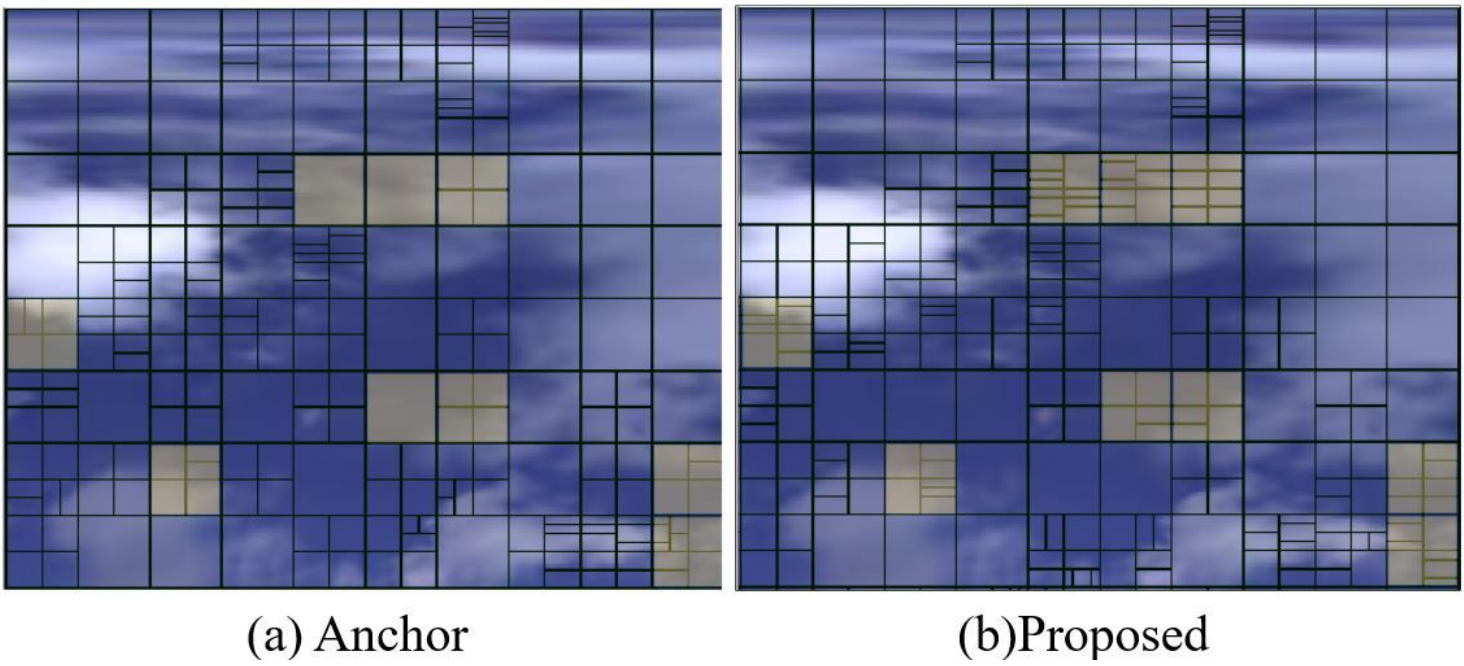

Figure 10

Comparisons of the block partitions between anchor (VTM4.0-360Lib-9.0) and the proposed scheme.

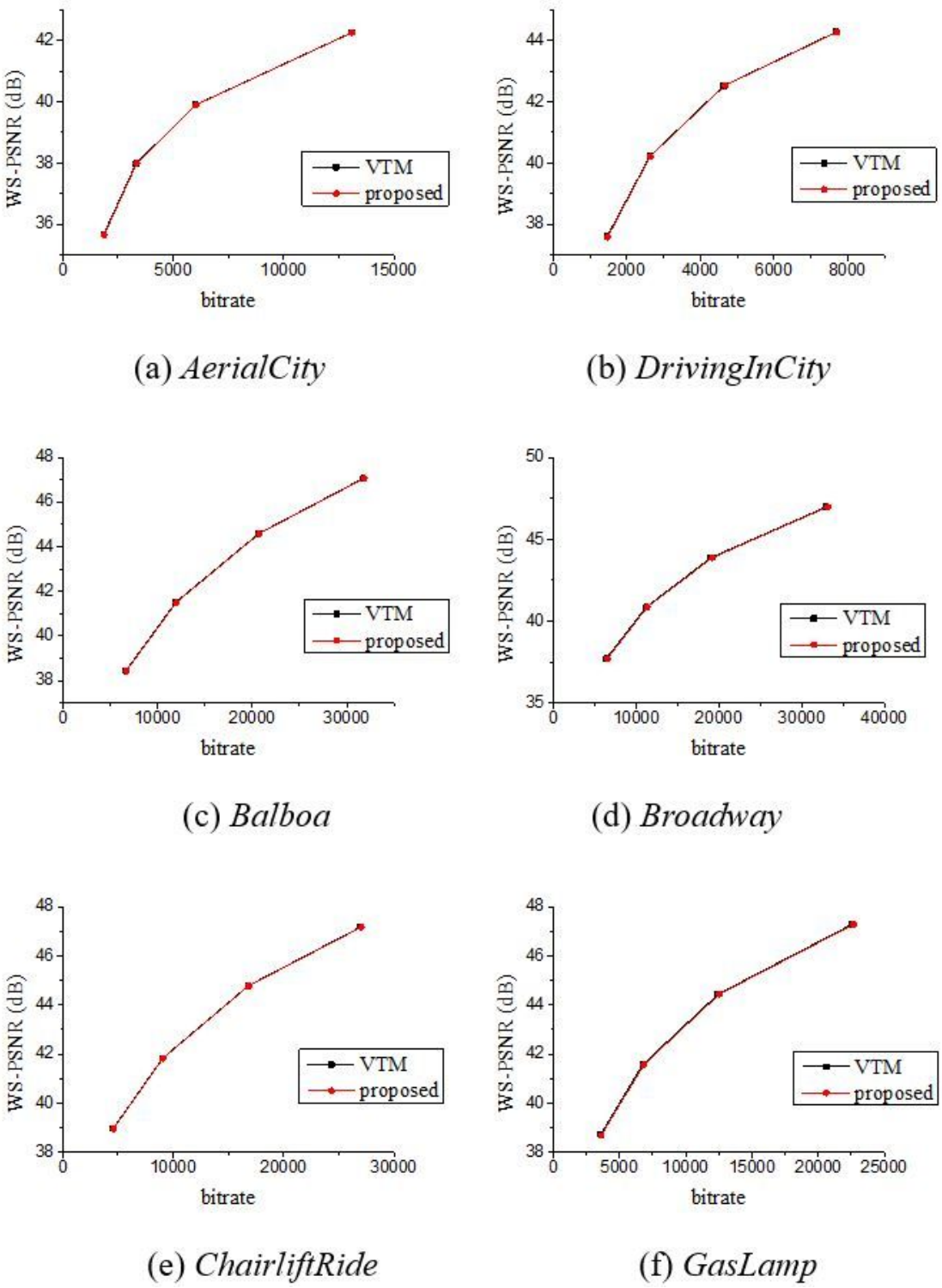

Figure 11

Performance comparison between the proposed algorithm and VTM4.0-360Lib-9.0.